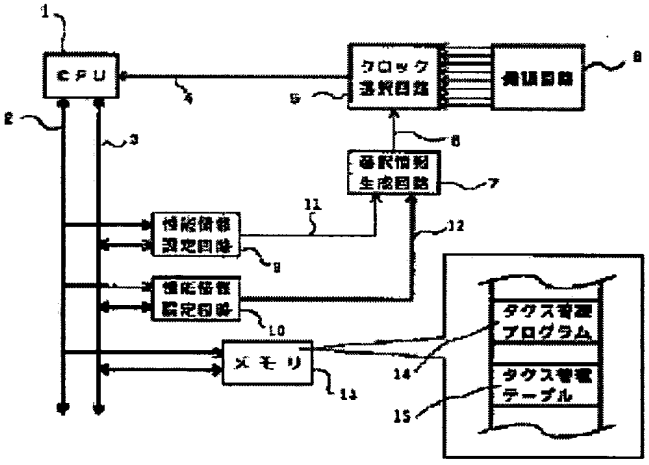
DEVICE AND METHOD FOR CONTROLLING CLOCK OF CENTRAL PROCESSING UNIT

Patent number: JP8076874
Publication date: 1996-03-22
Inventor: NISHIOKA KIYOKAZU; TANAKA KAZUHIKO; NOGUCHI YOSHIKI; OBA SHINYA
Applicant: HITACHI LTD
Classification:
- international: ***G06F1/04; G06F1/06; G06F9/46;*** **G06F1/04; G06F1/06; G06F9/46;** (IPC1-7): G06F1/04; G06F1/06; G06F9/46
- european:
Application number: JP19940212448 19940906
Priority number(s): JP19940212448 19940906

Abstract of **JP8076874**

PURPOSE: To control a clock of CPU so as to execute an operation with a low power consumption by automatically changing-over the clock into the absolute min. operation clock of CPU within a range where the requesting performance of a program to run is satisfied in the operation environment of a multi- task. CONSTITUTION: The clock controlling device is provided with a performance information of a central processing unit(CPU) 1, which is required at every task, and also provided with more than one performance information setting circuits 9 and 10 setting performance information of CPU 1 at every task, a selecting information generating circuit 7 deciding the clock frequency of CPU 1 so as to permit the operation to be executed with absolute min. performance which is required by the task in starting, an oscillation circuit 6 generating plural clock signals and a clock selecting circuit 5 selecting one of the clock signals and giving it to the CPU 1.



(19)日本国特許庁（ＪＰ）　(12)**公開特許公報（Ａ）**　(11)特許出願公開番号

特開平8－76874

(43)公開日　平成８年(1996)３月22日

(51)Int.Cl.[6]		識別記号	庁内整理番号	ＦＩ	技術表示箇所
Ｇ０６Ｆ	1/04	３０１　Ｃ			
	1/06				
	9/46	３４０　Ｚ	7737－5Ｂ		
				Ｇ０６Ｆ　1/04　３１０　Ａ	

審査請求　未請求　請求項の数７　ＯＬ　（全 15 頁）

(21)出願番号　特願平6－212448

(22)出願日　平成６年(1994)９月６日

(71)出願人　000005108
株式会社日立製作所
東京都千代田区神田駿河台四丁目６番地
(72)発明者　西岡　清和
神奈川県川崎市麻生区王禅寺1099　株式会社日立製作所システム開発研究所内
(72)発明者　田中　和彦
神奈川県川崎市麻生区王禅寺1099　株式会社日立製作所システム開発研究所内
(72)発明者　野口　孝樹
東京都国分寺市東恋ヶ窪一丁目280番地
株式会社日立製作所中央研究所内
(74)代理人　弁理士　高橋　明夫　（外１名）

最終頁に続く

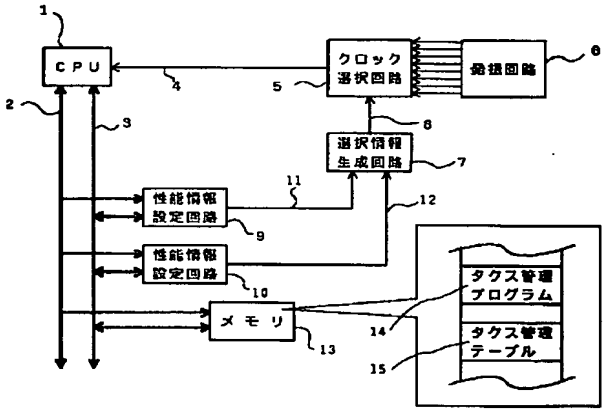
(54)【発明の名称】　中央処理装置のクロック制御装置およびクロック制御方法

(57)【要約】

【目的】マルチタスクの動作環境において、走行させるプログラムの要求性能を満足する範囲において、必要最低限のＣＰＵの動作クロックに自動的に切り替えて消費電力を節約し、低消費電力で作動するようにＣＰＵのクロックを制御する。

【構成】タスク毎に必要とする中央処理装置１の性能情報を設けて、前記タスク毎の中央処理装置１の性能情報を設定する一つ以上の性能情報設定回路９，１０と、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置１のクロック周波数を決定する選択情報生成回路７と、複数のクロック信号を発生する発振回路６と、そのクロック信号の中から一つを選択して前記中央処理装置１へ与えるクロック選択回路５を設ける。

図　1



1

【特許請求の範囲】

【請求項１】　複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、

前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御装置において、

前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、

前記タスク毎の中央処理装置の性能情報を設定する一つ以上の性能情報設定回路と、

前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成する選択情報生成回路と、

複数のクロック信号を発生する発振回路と、

前記選択情報に応じて、前記複数のクロック信号の中から一つを選択して前記中央処理装置へ与えるクロック選択回路を設けたことを特徴とする中央処理装置のクロック制御装置。

【請求項２】　前記中央処理装置と、前記一つ以上の性能情報設定回路と、選択情報生成回路と、クロック選択回路を１チップ内に集積することを特徴とする請求項１記載の中央処理装置のクロック制御装置。

【請求項３】　複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、

前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御方法において、

前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、

一つ以上の性能情報設定回路と、

選択情報生成回路と、

複数のクロック信号を発生する発振回路と、

クロック選択回路とを有し、

前記一つ以上の性能情報設定回路が、前記タスク毎の中央処理装置の性能情報を設定し、

前記選択情報生成回路が、前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成し、

前記クロック選択回路が、前記選択情報に応じて、前記

2

発振機から発生した複数のクロック信号の中から一つを選択して前記中央処理装置へ与えることを特徴とする中央処理装置のクロック制御方法。

【請求項４】　前記マルチタスクのオペレーティングシステムが、前記各タスクが必要とする性能情報を各タスク単位で管理し、

タスクを起動する際に、前記性能情報設定回路へ起動するタスクの性能情報を設定するステップを有することと、

さらに、タスクを終了する際に、前記性能情報設定回路に設定してある該当するタスクの性能情報を無効にするステップを有することとを特徴とする請求項３記載の中央処理装置のクロック制御方法。

【請求項５】　前記タスクを起動する際に、そのタスクの性能情報をタスク管理テーブルへ登録するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて、必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有すること、

さらに、タスクを終了する際に、そのタスクの性能情報をタスク管理テーブルから削除するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有することを特徴とする請求項４記載の中央処理装置のクロック制御方法。

【請求項６】　前記プログラムに従って、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成することを特徴とする請求項１および請求項２記載のいずれかの中央処理装置のクロック制御装置。

【請求項７】　前記情報処理装置が、電源供給手段として、電池とＡＣ電源のどちらでも使用でき、

電源供給手段の識別手段を設けることで、電池を使用しているときと、ＡＣ電源を使用しているときを識別し、その識別した結果によって、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成することを特徴とする請求項６記載の中央処理装置のクロック制御装置。

【発明の詳細な説明】

【０００１】

【産業上の利用分野】本発明は、中央処理装置のクロック制御装置およびクロック制御方法に係り、パーソナルコンピュータ（以下、単に「パソコン」という）や小形情報端末に代表される情報処理装置に使用される中央処理装置（Central Processing Unit、以下、「ＣＰＵ」と略記する）の省電力化を達成するのに好適な中央処理

3

装置のクロック制御装置およびクロック制御方法に関する。

【０００２】

【従来の技術】近年、電力需要の増加によって、社会全般にわたって電力の消費量が問題とされてきている。一方、このような状況下で、小形コンピュータ市場が急激に拡大し、パソコンの世界市場における設置台数は３００万台を越えている。また、電池で使用する携帯型情報機器の要求もたかまりつつあり、そのような機器の低電力化も重要な課題になりつつある。

【０００３】したがって、特に、このようなパソコンや携帯型情報機器などの情報処理装置の中枢であるＣＰＵの消費電力低減が注目されており、従来、多くの手法が試みられている。このようなＣＰＵの消費電力低減手法の有力なもののひとつとして、ＣＰＵのクロックを制御する手法がある。これは、特定の条件下において、低速なクロックでＣＰＵを動作させることにより、ＣＰＵの消費電力の抑制を達成するものである。ここで、この特定の条件は、電源電圧が任意のレベルまで低下したことや、ＣＰＵに対する割り込みが発生したことなどがある。

【０００４】さて、このようなＣＰＵの消費電力低減手法に関して、特に、情報端末など通信機能を実現する上で要求が強いマルチタスク機能を有する情報処理装置に適用する発明としては、特開昭６２－１５０４１６号公報に記載の「低消費電力状態への移行方式」がある。この発明は、複数のタスクを並行動作させるオペレーティングシステム（Operating System、以下、「ＯＳ」と略記する）を搭載したシステムにおいて、実行すべきタスクの有無を検出する手段と、コンピュータシステムを低消費電力状態にするの手段を設け、実行すべきタスクが無い場合にコンピュータシステムを低消費電力状態にする移行方式である。

【０００５】

【発明が解決しようとする課題】上記従来技術は、コンピュータシステムを低消費電力状態に移行する方式について述べている。しかしながら、上記従来技術は、実行すべきタスクが存在するかどうかを判定し、実行すべきタスクがない場合のみ、低消費電力状態で動作するものであり、タスク実行中に低消費電力状態で動作しないため、その適用範囲が狭いという問題点があった。

【０００６】また、近年は、多大な演算性能を要求するマルチメディア（動画、音声など）がシステムに取り込まれてくることが多くなっている。例えば、ワープロや表計算のソフトウエアと、テレビ会議のソフトウエアを一つのパソコンで動作させる場合が考えられる。ここで、前者のワープロや表計算のソフトウエアは、ＣＰＵが数１０ＭＩＰＳ（百万命令／毎秒、Million Instruction Per Second）の性能ならば十分使いものになるが、後者テレビ会議のソフトウエアは動画及び音声の圧縮伸

4

張処理に加えて通信機能も必要になり、数１００ＭＩＰＳの性能を必要とする。一方、ＣＰＵの性能は、年率約１．６倍程度の急速な高性能化傾向にあり、数年で数１００ＭＩＰＳの性能に達するものと予測できる。しかしながら、数１０ＭＩＰＳと数１００ＭＩＰＳの動作状態における消費電力の差は非常に大きいので、数１００ＭＩＰＳの性能を持つＣＰＵに数１０ＭＩＰＳのソフトウェアを動作させることは電力の無駄な消費である。

【０００７】したがって、使用するソフトウエアに応じて性能を自動的に切り換える要請があったが、従来技術では、プログラムの動作スピードに応じて、ＣＰＵの性能を切り替えるという考え方はされていないという問題点があった。

【０００８】本発明は、上記従来技術の問題点を解決するためになされたもので、その目的は、マルチタスクの動作環境において、その情報処理装置で動作させるプログラムの性能に応じて、低い性能で済む処理プログラムの実行時には、その要求性能を満足する必要最低限のＣＰＵの動作クロックに自動的に切り替えて消費電力を節約して、タスク実行中でも低消費電力での作動を実現しうるＣＰＵのクロック制御装置およびクロック制御方法を提供することである。

【０００９】また、本発明の他の目的は、その情報処理装置がＡＣ電源で作動するか電池で作動するかを判定し、電池で作動するときにのみ、低消費電力で作動するＣＰＵのクロック制御装置およびクロック制御方法を提供することである。

【００１０】

【課題を解決するための手段】上記目的を達成するために、本発明の中央処理装置のクロック制御装置に係る発明の構成は、複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御装置において、前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、前記タスク毎の中央処理装置の性能情報を設定する一つ以上の性能情報設定回路と、前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成する選択情報生成回路と、複数のクロック信号を発生する発振回路と、前記選択情報に応じて、前記複数のクロック信号の中から一つを選択して前記中央処理装置へ与えるクロック選択回路を設けたようにしたものである。

【００１１】より詳しくは、上記中央処理装置のクロッ

5

ク制御装置において、前記中央処理装置と、前記一つ以上の性能情報設定回路と、選択情報生成回路と、クロック選択回路を１チップ内に集積するようにしたものである。

【００１２】上記目的を達成するために、本発明の中央処理装置のクロック制御方法に係る発明の構成は、複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御方法において、前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、一つ以上の性能情報設定回路と、選択情報生成回路と、複数のクロック信号を発生する発振回路と、クロック選択回路とを有し、前記一つ以上の性能情報設定回路が、前記タスク毎の中央処理装置の性能情報を設定し、前記選択情報生成回路が、前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成し、前記クロック選択回路が、前記選択情報に応じて、前記発振機から発生した複数のクロック信号の中から一つを選択して前記中央処理装置へ与えるようにしたものである。

【００１３】より詳しくは、上記中央処理装置のクロック制御方法において、前記マルチタスクのオペレーティングシステムが、前記各タスクが必要とする性能情報を各タスク単位で管理し、タスクを起動する際に、前記性能情報設定回路へ起動するタスクの性能情報を設定するステップを有することと、さらに、タスクを終了する際に、前記性能情報設定回路に設定してある該当するタスクの性能情報を無効にするステップを有するようにしたものである。

【００１４】さらに詳しくは、上記中央処理装置のクロック制御方法において、前記タスクを起動する際に、そのタスクの性能情報をタスク管理テーブルへ登録するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて、必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有すること、さらに、タスクを終了する際に、そのタスクの性能情報をタスク管理テーブルから削除するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有するようにしたものである。

【００１５】また、上記目的を達成するために、本発明

6

の中央処理装置のクロック制御装置に係る発明の別の構成は、上記中央処理装置のクロック制御装置において、前記プログラムに従って、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成するようにしたものである。

【００１６】さらに別の構成は、上記上記中央処理装置のクロック制御装置において、前記情報処理装置が、電源供給手段として、電池とＡＣ電源のどちらでも使用でき、電源供給手段の識別手段を設けることで、電池を使用しているときと、ＡＣ電源を使用しているときを識別し、その識別した結果によって、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成するようにしたものである。

【００１７】

【作用】本発明によれば、個々のプログラムの固有の性能情報を有し、マルチタスク化においてタスクの起動および終了時に、動作しているプログラムの性能情報によって選択情報生成回路で必要なＣＰＵの性能を決定し、ＣＰＵのクロックを制御する。そのため、複数のプログラムが並列に実行している場合でも、各プログラムの要求性能を考慮してＣＰＵの動作クロックを決定しているので、必要かつ最低の電力で動作可能になる。

【００１８】また、電源検出回路によって、その情報処理装置の電源がＡＣ電源か電池かを判定し、電源が電池のときには選択情報生成回路で、ＣＰＵのクロックを低い状態で作動させることにしておくことにより、電池で作動するときにのみ、低消費電力で作動することになる。

【００１９】

【実施例】以下、本発明に係る各実施例を、図１ないし図１２を用いて説明する。

〔実施例１〕以下、本発明に係る第一の実施例を、図１ないし図６を用いて説明する。先ず、図１を用いて本発明に係るＣＰＵのクロック制御装置の回路構成について説明しよう。図１は、本発明の第一の実施例に係るＣＰＵのクロック制御装置の回路構成を示すブロック図である。

【００２０】図１において、発振回路６は、８種類の周波数のクロックを生成することができる。選択情報回路７は、クロック選択回路５を制御するための回路である。性能情報設定回路９，１０は、タスクの負荷情報を保持するための回路である。メモリ１３には、ＣＰＵ１が使用するプログラムやデータが格納される。そして、図に示される様に、タスク管理プログラム１４およびタスク管理テーブル１５は、このメモリ１３に格納されるものである。

【００２１】ＣＰＵ１へは、５～４０ＭＩＰＳまでＭＩＰＳ間隔で８レベルの動作状態を指定できるものとす

7

る。発振回路６は、上記各レベルに対応する８種類の周波数のクロックを出力する。

【００２２】また、本発明に係る情報処理装置は、マルチタスクのＯＳ (Operating System)が起動され、そのＯＳ上で各種プログラムを実行することを前提としている。ＯＳは、タスクの起動と終了を制御するタスク管理プログラム１４を含んでいる。さらに、タスク管理プログラム１４の制御に必要な各種情報がタスク管理テーブル１５に含まれている。具体的には、各タスクごとに、各タスクをユニークに識別するタスクＩＤ、必要なメモリ容量などに加えて、本発明に特徴的な性能情報とペンディングフラグが登録される項目として存在している。

【００２３】性能情報は、各タスクが実行するプログラムの内容に応じて指定されるＣＰＵの負荷となる能力である。ペンディングフラグは、あるタスクの性能情報が性能情報設定回路に設定するための待ち状態になっていることを示すフラグである。

【００２４】ペンディングフラグの役割は、後にフローを用いて本発明の動作を説明するときに明らかになるので、先に、性能情報について説明する。性能情報は、上で述べたように、タスクが実行するプログラムの内容に応じて指定する。例えば、ワードプロセッサに関するプログラムの場合で、編集プログラムは、１０ＭＩＰＳ、印刷プログラムは、５ＭＩＰＳのＣＰＵ性能が必要であるとする。

【００２５】このような状況において、タスク管理プログラム１４が、編集プログラムのタスクを起動する際に、性能情報設定回路９へ１０ＭＩＰＳの性能情報を設定すると、選択情報回路７は、１０ＭＩＰＳに相当するＣＰＵ１の動作周波数に対応する選択情報を生成する。この選択情報にしたがって、クロック選択回路５は該当するクロックを選択して、クロック線４を介して、ＣＰＵ１へ供給する。これにより、ＣＰＵ１は１０ＭＩＰＳで動作する。これが、本発明の基本的な仕組みである。

【００２６】続いて、ワードプロセッサで編集した文書を印刷する際には、タスク管理プログラム１４が印刷プログラムのタスクを起動する。このとき、性能情報設定回路１０へ5ＭＩＰＳの性能情報を設定する。選択情報回路７は、性能情報設定回路９に設定した１０ＭＩＰＳの性能情報と性能情報設定回路１０に設定した性能情報から、１５ＭＩＰＳに相当するＣＰＵ１の動作周波数に対応する選択情報を生成する。この選択情報にしたがって、クロック選択回路５は該当するクロックを選択して、クロック線４を介して、ＣＰＵ１へ供給する。これにより、ＣＰＵ１は１０ＭＩＰＳから１５ＭＩＰＳへ、より高速な動作モードへ移行する。

【００２７】性能情報は、時間ごとの処理能力であらわされるので、両タスクが並列実行されるばあいには、両タスクの加算になることに留意しよう。

【００２８】本発明によれば、このように、ワードプロセッサで編集中に印刷を実行した場合でも、負荷に見合ったようにＣＰＵ１の性能を向上させて、印刷しながらでも編集のための十分な操作環境を得ることができるのである。

8

【００２９】次に、図２を用いて性能情報設定回路の詳細な回路構成について説明しよう。図２は、本発明の第一の実施例に係る性能情報設定回路の回路構成を示すブロック図である。

【００３０】上述したように、性能情報設定回路９は、タスク管理プログラム１４によって起動される各タスクごとの性能情報を保持するための回路である。タスク起動時に、性能情報が性能情報設定レジスタ２２へ設定されると共に、そのタスクのタスクＩＤがタスクＩＤ設定レジスタ２０へ、さらに、タスクＩＤ設定レジスタ２０と性能情報設定レジスタ２２の設定が有効であることを示すイネーブル情報がイネーブル設定レジスタ２１へ設定される。イネーブル設定レジスタ２１は、論理値「１」の時に有効状態、論理値「０」の時に無効状態を示す。したがって、論理積回路２３は、有効状態の場合のみ、性能情報設定レジスタ２２の情報を性能情報線１１へ出力する。逆に、無効状態の場合、論理値「０」が性能情報線１１へ出力される。

【００３１】ここで、イネーブル設定レジスタ２１が、有効状態のときが、この性能情報設定回路に設定された値が有効であることを示し、逆の無効状態が、この性能情報設定回路に設定された値が無効であり、いわば、値が設定されていないことを示している。

【００３２】また、タスクＩＤ設定レジスタ２０、イネーブル設定レジスタ２１および性能情報設定レジスタ２２の設定情報は、アドレスバス２およびデータバス３を介して、ＣＰＵ１が読みだし可能であり、タスク管理プログラム１４が性能情報設定回路９の設定情報を知ることができる。

【００３３】次に、図３を用いて選択情報生成回路の詳細な回路構成について説明しよう。図３は、本発明の第一の実施例に係る選択情報生成回路の回路構成を示すブロック図である。

【００３４】本実施例は、ＣＰＵ１へ８レベルの動作状態を指定できたことを想起しよう。したがって、この選択情報生成回路７は、３ビット情報を扱うものになる。

【００３５】図３において、加算回路３０は、３ビットの情報を加算する。キャリー信号線３１は、加算結果が桁上がりした時に論理値「１」を示す。デコーダ回路３５は、３ビットである。

【００３６】加算回路３０は、性能情報線１１の性能情報（０～７）と性能情報線１２の性能情報（０～７）の加算結果を、論理和回路３２～論理和回路３４へ出力する。加算結果が７以下の場合、キャリー信号線３１が論理値「０」となり、論理和回路３２～論理和回路３４は、加算回路３０の加算結果をそのままデコーダ回路３

9

５へ出力する。

【００３７】一方、加算回路３０の加算結果が８以上の場合、加算結果７がデコーダ回路３５へ出力される。要するに、キャリー信号線３１が論理値「１」となり、論理和回路３２～論理和回路３４は全て論理値「１」をデコーダ回路３５へ出力する。このような場合分けは、クロック選択回路５が選択できるクロックが８種類に限られているために、性能情報の最高のレベルを７に押さえるため必要となるものである。例えば、クロック選択回路５が１６種類のクロックを選択可能ならば、この選択情報回路の構成も異なったものになる。

【００３８】このように論理和回路３２～論理和回路３４が出力する性能情報を受けて、デコーダ回路３５は、該当するクロック信号を選択するための情報を選択情報信号線８へ出力する。

【００３９】以上説明したように、選択情報回路７は、性能情報線１１および１２の情報から生成したクロック選択情報を、選択情報信号線８へ出力し、クロック選択回路５に適切なクロックを選択させるものである。

【００４０】次に、図４および図５を用いてタスク管理プログラム１４の詳細を説明しよう。図４は、タスク起動時のタスク管理プログラムの動作をあらわすフローチャートである。図５は、タスク終了時のタスク管理プログラムの動作をあらわすフローチャートである。

【００４１】先ず、図４を用いてタスク管理プログラム１４がタスクを起動する場合の動作を、図の順を追って説明しよう。最初に、通常のマルチタスクＯＳがおこなう所定のタスク起動処理がおこなわれる（Ｓ４００）。ここでは、タスク管理テーブル１５の内容が更新される。すなわち、タスク管理テーブル１５に、新しく起動するタスクのタスクＩＤ、必要なメモリ容量、性能情報などが登録される。

【００４２】次に、性能情報設定回路９，１０に設定されているイネーブル情報を読み込み（Ｓ４０１）、その情報が有効か無効かをチェックする（Ｓ４０２）。これは、使われていない性能情報設定回路があるか調べるものである。

【００４３】どちらかのイネーブル情報が無効状態ならば、該当する方の性能情報設定回路（９，１０のどちらか）へ、起動するタスクのタスクＩＤ、イネーブル有効情報及び性能情報を設定する（Ｓ４０３）。これにより、新しいタスクが起動された環境下において最適なクロック周波数でＣＰＵ１が動作することになり、タスク起動処理を終了する。

【００４４】一方、Ｓ４０２のステップにおいて、イネーブルが全て有効状態ならば、性能情報設定回路は、すべて他のタスクに使用されている状態である。この場合には、新しく起動したタスクによって、これらの性能情報設定回路９，１０の性能情報を更新するべきかどうかを調べる必要がある。

10

【００４５】この場合は、これらの性能情報設定回路９，１０の性能情報を読み込む（Ｓ４０４）。

【００４６】次に、読み込んだ各々の性能情報と、起動するタスクの性能情報を、比較する（Ｓ４０５）。その結果、起動するタスクの性能情報が最も低い値であるならば、既に起動されているタスクによって、十分にＣＰＵの性能が上昇している状態である。よってこの場合は、性能情報設定回路９，１０の更新せずに、タスク管理テーブル１５中の起動するタスクのペンディングフラグを有効状態にして（Ｓ４０７）、終了する。ここで、ペンディングフラグは、タスクとしては、起動したものの、そのことが性能情報設定回路９，１０には、影響を与えていないことを示している。このペンディングフラグは、後のタスク終了時に参照される。

【００４７】一方、起動するタスクの性能情報が最も低い値でないならば、この起動されたタスクに対応して、ＣＰＵの性能を上昇させなければならない。したがって、この場合には、最も低い値が設定されている方の性能情報設定回路（９，１０のどちらか）へ、起動するタスクのタスクＩＤ、イネーブル有効情報及び性能情報を設定する（Ｓ４０６）。そして、その書換えた性能情報設定回路のプロセスＩＤに対応するプロセスのペンディングフラグを有効にして、終了する。これは、書換えにより、既に起動されているタスクが、性能情報設定回路の性能に反映されなくなったため、後で復帰させるためである。

【００４８】これにより、新しいタスクが起動された環境下において最適なクロック周波数でＣＰＵ１が動作することになり、タスク起動処理を終了する。

【００４９】以上説明したように、本実施例は、２個の性能情報設定回路９と１０で構成しているが、この個数を増やす場合にも対応可能であり、個数を増やすほどきめ細かなクロック制御が可能になり、省電力化の効果を得ることができる。

【００５０】先ず、図５を用いてタスク管理プログラム１４がタスクを終了させる場合の動作を、図の順を追って説明しよう。最初に、性能情報設定回路９および１０に設定されているタスクＩＤ情報を読み込み（Ｓ５０１）、その情報が終了しようとするタスクのタスクＩＤと一致しているかをチェックする（Ｓ５０２）。これは、終了しようとするタスクの性能が性能情報設定回路に反映されているか調べるためのものである。

【００５１】どちらかの性能情報設定回路に格納されたタスクＩＤと終了しようとするタスクのタスクＩＤ情報が一致するならば、該当する方の性能情報設定回路（９，１０のどちらか）へ、イネーブル無効情報を設定する（Ｓ５０３）。これは、性能情報設定回路の情報を消去したことに該当する。

【００５２】次に、タスク管理テーブル１５中にペンディングフラグが有効状態となっているタスクが存在する

11

かをチェックする（Ｓ５０４）。ペンディングフラグが有効状態となっているということは、そのタスクが起動されており、性能情報設定回路９，１０に性能を設定するために待ち状態になっていると考えることができる。

したがって、ペンディングフラグが有効状態となっているタスクが存在するならば、そのタスクのタスクＩＤ、イネーブル有効情報及び性能情報を、ステップ５０３において該当した性能情報設定回路（９，１０のどちらか）へ設定する（Ｓ５０５）。そして、設定したタスクのペンディングフラグを無効状態にする（Ｓ５０６）。

【００５３】これにより、タスクが終了した環境下において最適なクロック周波数でＣＰＵ１が動作することになる。

【００５４】最後に、タスク管理テーブル１５から終了するタスクに関する情報を削除するなど、通常のマルチタスクＯＳが行う所定のタスク終了処理がおこなわれ（Ｓ５０７）、タスク終了処理を終了する。

【００５５】一方、ステップＳ５０２において、どちらかのタスクＩＤ情報も一致しないならば、終了するタスクはＣＰＵ１の動作周波数決定に影響していないことになる。したがって、ステップＳ５０３～Ｓ５０６を飛び越して、ステップＳ５０７を実行し、タスク終了処理を終了する。

【００５６】同様に、ステップＳ５０４において、ペンディングフラグが有効状態のタスクが存在しないならば、ＣＰＵ１の動作周波数決定に影響するべきタスクがないことになる。したがって、ステップＳ５０５、Ｓ５０６を飛び越して、ステップ５０７を実行し、タスク終了処理を終了する。

【００５７】最後に、図６を用いて以上説明した実施例の具体的な動作の例を経時順に説明してみよう。図６は、各タスクの状態と性能情報の関係を経時順に示したタイミングチャートである。

【００５８】より詳しくは、図６は、各タスクの起動と終了を示すイベントと、起動されたタスクの状態（実行状態と待機状態間の遷移）と、性能情報設定回路９および性能情報設定回路１０に設定される性能情報と、ＣＰＵ１の実動作に相当する性能情報を示したものである。この図における時間軸の単位として、各イベントを起点とした６つのタイムスロット（０～５）を用いることにした。

【００５９】図６の様に、タスクＡ、タスクＢ、タスクＣが起動されるのであるが、これらの各々に対応する性能情報は、「２」、「４」、「５」とすることにしよう。

【００６０】先ず、タイムスロット０では、ＯＳだけが動作しており、実行中のタスクは存在しない。このとき、性能情報設定回路９および性能情報設定回路１０にはイネーブルが無効状態に設定されており、ＣＰＵ１

12

は、最低性能「０」で動作している。

【００６１】次に、タスクＡが起動されたタイムスロット１では、タスク管理プログラム１４が性能情報設定回路９へイネーブル有効情報と性能情報「２」を設定する。これにより、ＣＰＵ１は性能情報「２」に相当する性能で動作することになる。

【００６２】次に、タスクＢが起動されたタイムスロット２では、先ず、タスクＢが実行状態となり、タスクＡは実行状態から待機状態へ遷移する。さらに、タスク管理プログラム１４が性能情報設定回路１０へイネーブル有効情報と性能情報「４」を設定する。これにより、性能情報設定回路９に設定された性能情報「２」と性能情報設定回路１０に設定された性能情報「４」が、選択情報生成回路７で加算されて、ＣＰＵ１は性能情報「６」に相当する性能で動作する。この動作環境下で、タスクＡとタスクＢは、背反的に実行状態と待機状態の間を遷移する。

【００６３】次に、タスクＣが起動されたタイムスロット３では、先ず、タスクＣが実行状態となり、タスクＡは実行状態から待機状態へ遷移する。起動されるタスクＣの性能情報は、「５」なので、タスク管理プログラム１４は、性能情報設定回路９および性能情報設定回路１０へ設定されている性能情報を比較し、低い性能情報が設定されている方の性能情報設定回路９へ、性能情報「５」を設定する。これにより、本来性能情報は「９」となるが、本実施例では、性能情報の最大レベルが「７」なので、ＣＰＵ１は性能情報「７」に相当する性能で動作する。

【００６４】また、タスク管理プログラム１４は、タスク管理テーブル１５中のタスクＡのペンディングフラグを有効状態とする。このような動作環境下で、タスクＡ、タスクＢ、タスクＣは、背反的に実行状態と待機状態の間を遷移する。

【００６５】次に、タスクＢが終了されたタイムスロット４では、先ず、タスクＣが実行状態となり、タスクＢは終了する。さらに、タスク管理プログラム１４は、タスクＢの性能情報「４」の代わりにペンディングフラグが有効状態となっているタスクＡの性能情報「２」を性能情報設定回路１０へ設定する。すなわち、追い出されていたタスクＡの性能情報を復帰させるわけである。これにより、ＣＰＵ１は性能情報「７」に相当する性能で動作する。この動作環境下で、タスクＡとタスクＣは、背反的に実行状態と待機状態の間を遷移する。

【００６６】最後に、タスクＡが終了されたタイムスロット５では、先ず、タスクＣが実行状態となり、タスクＡは終了する。さらに、タスク管理プログラム１４は、終了するタスクＡの性能情報「２」を無効にするため性能情報設定回路１０へイネーブル無効情報を設定する。これにより、性能情報設定回路９だけが有効となり、ＣＰＵ１は性能情報「５」に相当する性能で動作する。

13

【００６７】以上説明した第一の実施例では、性能情報設定回路が２個の構成であるが、これに限定したわけではなく、性能情報設定回路の数を増やせば、ＣＰＵ１の動作速度を、さらに、きめ細かく制御できる。同様に、ＣＰＵ１のクロック周波数も８種類に限定したわけではなく、周波数の選択しを増やせば、さらにきめ細かな省電力制御が可能になる。

【００６８】〔実施例２〕以下、本発明に係る第二の実施例を、図７を用いて説明する。図７は、本発明の第二の実施例に係るＣＰＵのクロック制御装置の回路構成を示すブロック図である。

【００６９】この第二の実施例は、基本的な構成と動作およびその思想は、同様のものによるものであるが、その特徴は、回路構成を集積して１チップ化するところにある。

【００７０】発振器６２は、クロック信号を発生し、分周回路６１は、周波数が異なる８種類のクロック信号を発生する。また、低電力対応ＣＰＵ６０は、ＣＰＵ１と、性能情報設定回路９，１０と、選択情報生成回路７と、クロック選択回路５と、分周回路６１とを１チップに集積化したプロセッサである。

【００７１】この実施例では、性能情報設定回路９などの回路部が低電力対応ＣＰＵ６０に集積化されておるため、回路構成全体の部品点数を削減できるという利点がある。

【００７２】また、低電力対応ＣＰＵ６０が分周回路６１を内蔵するため、発振器６２からのクロック信号線は１本で済むことになる。低電力対応ＣＰＵ６０に内蔵する回路は、第一の実施例で説明したように、比較的簡単なハードウエアで構成できるため、回路規模が比較的小さく、ピン数もＣＰＵ１と比較して、それほど増加することはないので、十分に集積回路として構成することは可能である。

【００７３】この第二の実施例の効果としては、回路を集積化することによる小形化、低電力化だけでなく、ハードウエアの実装設計が容易になることもある。すなわち、クロック選択回路５をチップ内部に取り込んだことで、ＣＰＵクロックの高速化に伴い問題となるクロック信号の反射や干渉など実装上の技術課題が緩和されるのである。

【００７４】〔実施例３〕以下、本発明に係る第三の実施例を、図８ないし図１０を用いて説明する。図８は、本発明の第三の実施例に係るＣＰＵのクロック制御装置の回路構成を示すブロック図である。

【００７５】この第三の実施例の特徴は、第一の実施例において、性能情報設定回路９，１０と選択情報設定回路７が生成する選択情報を、図８に示されるタスク管理プログラム７１に従って、選択情報設定回路７０が生成することにある。

【００７６】ここで、選択情報設定回路７０は、クロック選択回路５へ与える選択情報を設定する回路であり、タスク管理プログラム７１は、選択情報を生成する機能を持つプログラムである。

14

【００７７】以下、この選択情報設定回路７０の機能を、図９を用いて詳細に説明しよう。図９は、本発明の第三の実施例に係る選択情報設定回路の回路構成を示すブロック図である。

【００７８】本実施例でも、取扱える性能レベルは、８段階を想定しており、したがって、レジスタ８０は、３ビットである。選択情報設定回路７０においては、アドレスバス２とデータバス３を介して、ＣＰＵ１が３ビットの性能情報を設定するとともに、設定した性能情報を読み出すことができる。設定する性能情報は、実施例１とは異なり、性能情報設定回路が生成するのではなく、タスク管理プログラム７１によって生成する。この性能情報は、選択情報設定回路７０の中のデコーダ回路３５へ送られ、デコーダ回路３５は、上記性能情報からクロック選択情報を生成して、選択情報信号線８へ出力し、これによって、最終的にＣＰＵ１の動作周波数が決められることになる。

【００７９】次に、実施例１との相違も考慮に入れて、タスク管理プログラム７１の性能情報を設定する動作の詳細を、図１０の順を追って説明しよう。図１０は、タスク管理プログラムの性能情報を設定する動作をあらわすフローチャートである。

【００８０】先ず、タスク管理プログラム７１は、これからおこなう処理がタスクの起動処理か終了処理かをチェックする（Ｓ９００）。

【００８１】起動処理ならば、図４のステップＳ４００と同様に、所定のタスク生成処理を行う（Ｓ９０１）。ここでは、起動するタスクに対応する各種情報をタスク管理テーブル１５へ登録する。

【００８２】逆に、終了処理ならば、図５のステップＳ５０７と同様に、所定のタスク終了処理を行う（Ｓ９０２）。ここでは、終了するタスクに対応する各種情報をタスク管理テーブル１５から削除する。

【００８３】ステップＳ９０１およびＳ９０２の次には、起動中のタスクが存在するかをチェックする（Ｓ９０３）。このチェックは、タスク管理テーブル１５への登録の有無を調べれば良い。

【００８４】登録があるならば、登録されている全タスクの性能情報を読み出す（Ｓ９０４）。次に、読みだした性能情報の総和を求めて、ＣＰＵ１の性能情報を生成し（Ｓ９０５）、ＣＰＵ１の性能情報の値が「８」を越えているかをチェックする（Ｓ９０６）。越えてないならば、そのＣＰＵ１の性能情報をレジスタ８０へ設定する（Ｓ９０７）。

【００８５】逆に、ＣＰＵ１の性能情報の値が「８」を越えているならば、ＣＰＵ１の性能情報の値を「７」（本実施例における性能情報の最大値）として（Ｓ９０

15

８）、レジスタ８０へ設定する（Ｓ９０７）。

【００８６】また、ステップＳ９０３において、起動中のタスクが存在しないならば、ＣＰＵ１の性能情報の値を「０」（本実施例における性能情報の最小値）として（Ｓ９０９）、レジスタ８０へ設定する（Ｓ９０７）。

【００８７】このように、タスクの起動時および終了時において、性能情報からＣＰＵ１へ与える選択情報を生成する機能をタスク管理プログラム７１に持たせれば、この既脳をソフトウエアで実現できる。したがって、本実施例においては、実施例１とは異なり性能情報設定回路９，１０が不要であり、ハードウエアの部品点数を削減できるという利点がある。

【００８８】〔実施例４〕以下、本発明に係る第四の実施例を、図１１および図１２を用いて説明する。本実施例の特徴は、使用している電源に従って、ＣＰＵ１の性能を制御することにある。

【００８９】それを考慮して、図１１を用いて、本実施例に係るＣＰＵのクロック制御装置の回路構成と電源回路の構成について説明しよう。図１１は、本発明の第四の実施例に係るＣＰＵのクロック制御装置の回路構成と電源回路の構成を示すブロック図である。

【００９０】選択情報生成回路１００は、性能情報からクロック信号の選択情報を生成することができる。電源検出回路１１０は、電源供給手段を識別するための回路である。また、電源制御回路１１１は、電源供給手段をＡＣ電池１１３にするか電池１１４にするかを制御する。電源供給手段の選択回路１１２は、実際にどちらの電源を採用するか切り替える回路である。ここで、ＡＣ電源１１３は、このＣＰＵを用いた情報処理装置の外部から電力供給され、電池１１４は、情報処理装置に内蔵されることを想定している。これらＡＣ電源１１３と電池１１４は、どちらか一方が電源供給手段として使用されるものであり、どちらを使用するかは、電源制御回路１１１によって決定される。電源制御回路１１１の指示

16

により、選択回路１１２はＡＣ電源１１３と電池１１４のいずれかを選択して情報処理装置で電力を供給するものである。

【００９１】また、電源検出回路１１０は、電源制御回路１１１が選択回路１１２へ指示した情報がＣＰＵ１によって読み出されることを可能にし、ＣＰＵ１が現在どちらの電源供給手段を使用中であるかを検出できる。

【００９２】選択情報生成回路１００がＣＰＵ１の性能情報を選択するのは、ＣＰＵ１が情報を設定するアドレスバス２とデータバス３を介して入力されるデータにより決定される。

【００９３】以下、図１２を用いて、この選択情報回路１００の構成と動作について詳細に説明しよう。図１２は、本発明の第四の実施例に係る選択情報回路の回路構成を示すブロック図である。

【００９４】２ポートＲＡＭ１０１は、アドレス４ビット、データ３ビット、デコーダ回路１０２は３ビットとして構成されている。なお、本実施例も性能レベルは、「０」から「７」までの、８段階を想定している。２ポートＲＡＭ１０１は、アドレスバス２およびデータバス３から情報が設定される。この設定された情報にしたがって、性能情報線１１，１２の４ビットの性能情報がアドレス情報となり、その結果として読み出された３ビットのデータがデコーダ回路１０２へ送られる。送られたデータから、デコーダ回路１０２は、８ビットのクロック選択情報を出力し、８種類クロック信号のうち１つを選択する。

【００９５】ここで、次の表１と表２を用いて、この２ポートＲＡＭ１０１へ設定される情報と出力の例を説明しよう。表１は、ＡＣ電源１１３を使用している通常動作モード時の性能情報の設定を対照した表である。

【００９６】

【表１】

17　　18

表　1

アドレス（性能情報）				データ（設定情報）			
信号線11		信号線12		クロックレベル	信号線８		
1	1	1	1	7	1	1	1
1	1	1	0	6	1	1	0
1	1	0	1	5	1	0	1
1	1	0	0	4	1	0	0
1	0	1	1	5	1	0	1
1	0	1	0	4	1	0	0
1	0	0	1	3	0	1	1
1	0	0	0	2	0	1	0
0	1	1	1	4	1	0	0
0	1	1	0	3	0	1	1
0	1	0	1	2	0	1	0
0	1	0	0	1	0	0	1
0	0	1	1	3	0	1	1
0	0	1	0	2	0	1	0
0	0	0	1	1	0	0	1
0	0	0	0	0	0	0	0

【００９７】表２は、電池１１４を使用している省電力動作モード時の設定対照した表である。

＊【００９８】

＊20　【表２】

表　2

アドレス（性能情報）				データ（設定情報）			
信号線11		信号線12		クロックレベル	信号線８		
1	1	1	1	5	1	0	1
1	1	1	0	5	1	0	1
1	1	0	1	5	1	0	1
1	1	0	0	4	1	0	0
1	0	1	1	5	1	0	1
1	0	1	0	4	1	0	0
1	0	0	1	3	0	1	1
1	0	0	0	2	0	1	0
0	1	1	1	4	1	0	0
0	1	1	0	3	0	1	1
0	1	0	1	2	0	1	0
0	1	0	0	1	0	0	1
0	0	1	1	3	0	1	1
0	0	1	0	2	0	1	0
0	0	0	1	1	0	0	1
0	0	0	0	0	0	0	0

【００９９】表１に示される性能情報の設定は、基本的に第一の実施例と同様のアルゴリズムによるものであり、性能情報線１１と１２の情報を加算するアルゴリズムである。したがって、実行しているタスクの性能情報の和が実際のＣＰＵ１のクロックレベルとして用いられる。

【０１００】一方、表２に示される性能情報の設定は、電池駆動で使用している場合に用いられる場合のもので、ＣＰＵ１がフルパワーで動作しないようなアルゴリズムである。こりアルゴリズムでは、ＣＰＵ１が高速で動作するクロックレベル６と７の設定を使用しない、つまり、性能情報線１１と１２の総和が６以上の時はクロックレベル５に設定するように工夫されている。

【０１０１】このように、第四の実施例では、電源供給手段の使用状態に応じて、ＣＰＵ１へ与えるクロック周波数を制御することで、電池１１４を用いている場合は、処理性能は低下するものの、使用している電池の動作時間を長くできる効果がある。

【０１０２】

【発明の効果】本発明によれば、マルチタスクの動作環境において、その情報処理装置で動作させるプログラムの性能に応じて、低い性能で済む処理プログラムの実行時には、その要求性能を満足する必要最低限のＣＰＵの動作クロックに自動的に切り替えて消費電力を節約し

19

て、タスク実行中でも低消費電力での作動を実現しうるＣＰＵのクロック制御装置およびクロック制御方法を提供することができる。

【０１０３】また、本発明によれば、その情報処理装置がＡＣ電源で作動するか電池で作動するかを判定し、電池で作動するときにのみ、低消費電力で作動するＣＰＵのクロック制御装置およびクロック制御方法を提供することができる。

【図面の簡単な説明】

【図１】本発明の第一の実施例に係るＣＰＵのクロック制御装置の回路構成を示すブロック図である。

【図２】本発明の第一の実施例に係る性能情報設定回路の回路構成を示すブロック図である。

【図３】本発明の第一の実施例に係る選択情報生成回路の回路構成を示すブロック図である。

【図４】タスク起動時のタスク管理プログラムの動作をあらわすフローチャートである。

【図５】タスク終了時のタスク管理プログラムの動作をあらわすフローチャートである。

【図６】各タスクの状態と性能情報の関係を経時順に示したタイミングチャートである。

【図７】本発明の第二の実施例に係るＣＰＵのクロック制御装置の回路構成を示すブロック図である。

【図８】本発明の第三の実施例に係るＣＰＵのクロック制御装置の回路構成を示すブロック図である。

20

【図９】本発明の第三の実施例に係る選択情報設定回路の回路構成を示すブロック図である。

【図１０】タスク管理プログラムの性能情報を設定する動作をあらわすフローチャートである。

【図１１】本発明の第四の実施例に係るＣＰＵのクロック制御装置の回路構成と電源回路の構成を示すブロック図である。

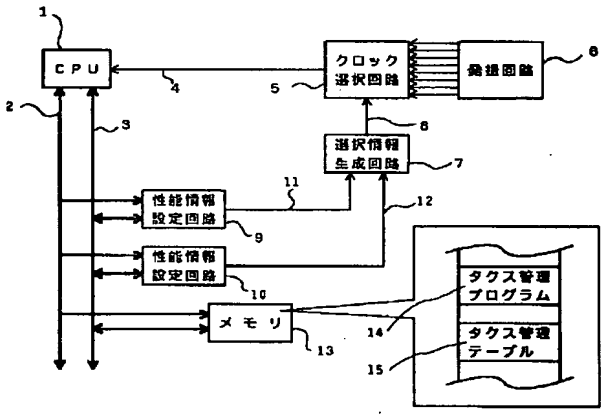
【図１２】本発明の第四の実施例に係る選択情報回路の回路構成を示すブロック図である。

【符号の説明】

１…ＣＰＵ、２…アドレスバス、３…データバス、４…ＣＰＵ１のクロック線、５…クロック選択回路、６…発振回路、７…選択情報生成回路、８…選択情報信号線、９，１０…性能情報設定回路、１１，１２…性能情報線、１３…メモリ、１４…タスク管理プログラム、１５…タスク管理テーブル。２０…タスクＩＤ設定レジスタ、２１…イネーブル設定レジスタ、２２…性能情報設定レジスタ、２３…論理積回路。３０…加算回路、３１…キャリー信号線、３２～３４…論理和回路、３５…デコーダ回路。６０…ＣＰＵ、６１…分周回路、６２…発振器。７０…選択情報設定回路、７１…タスク管理プログラム。８０…レジスタ。１００…選択情報生成回路、１１０…電源検出回路、１１１…電源制御回路、１１２…選択回路、１１３…ＡＣ電源、１１４…電池。１０１…２ポートＲＡＭ、１０２…デコーダ回路。

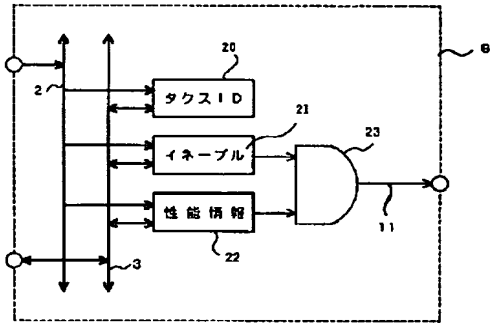
【図１】

図　1



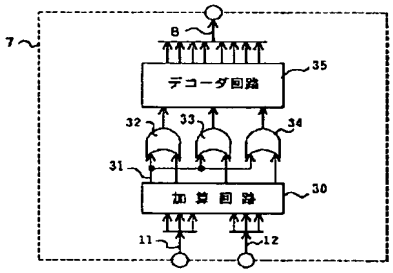
【図２】

図　2



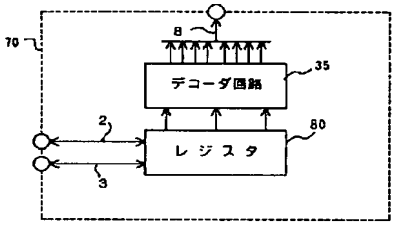
【図３】

図　3



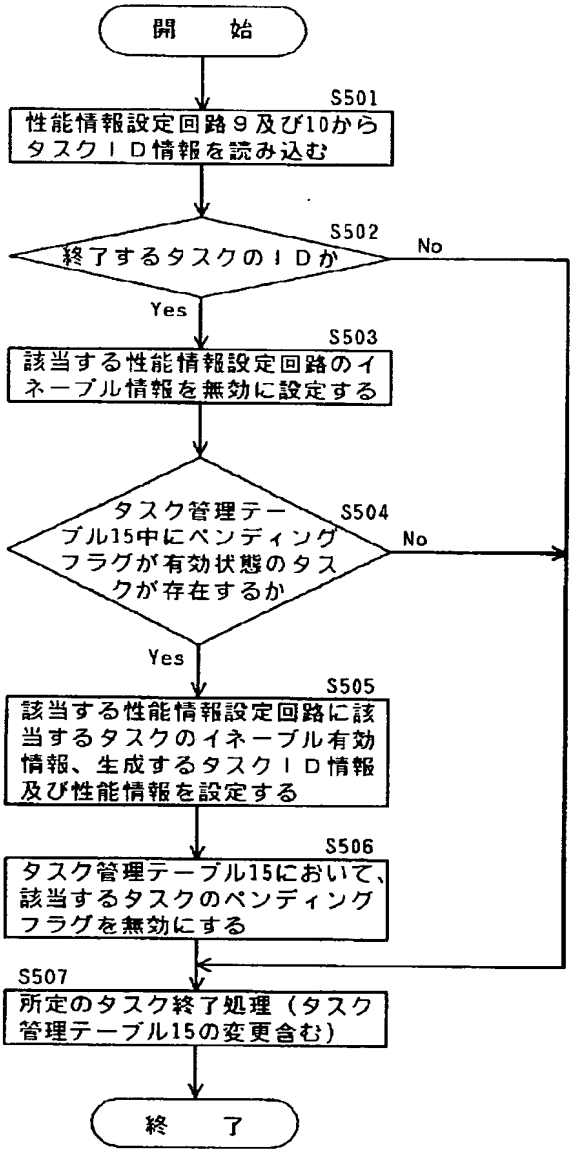
【図９】

図　9



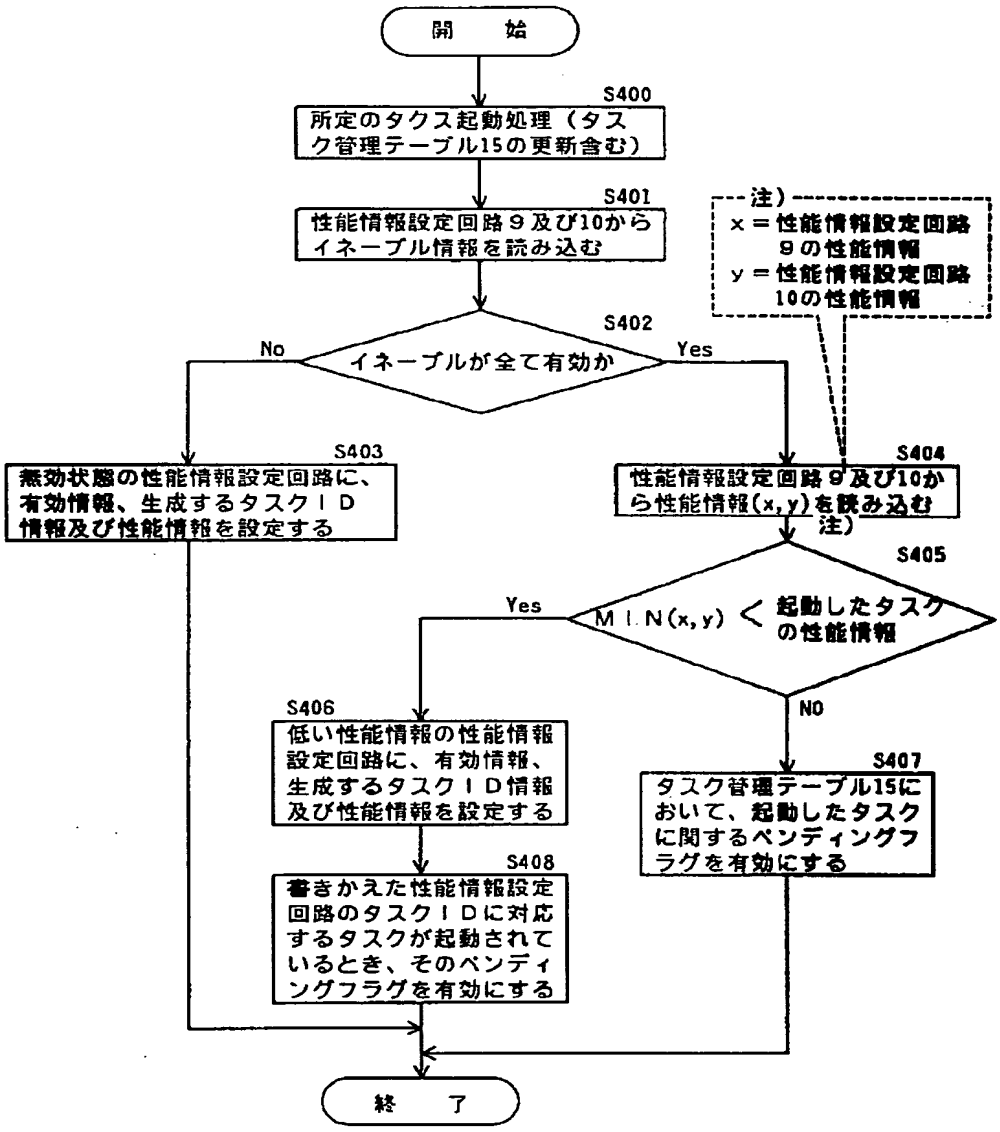
【図５】

図　5



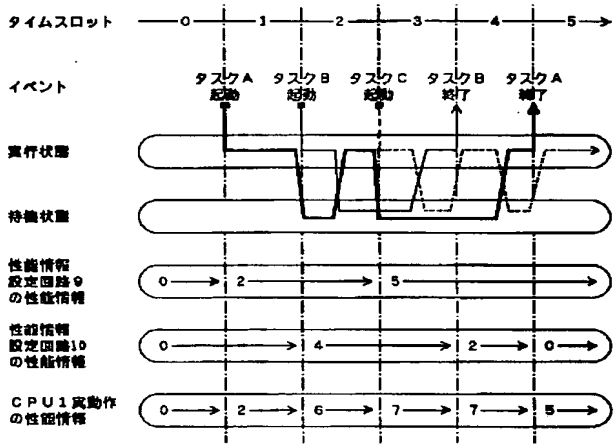
【図４】

図　４


開　始
S400
所定のタクス起動処理（タスク管理テーブル15の更新含む）
S401
性能情報設定回路９及び10からイネーブル情報を読み込む
注）
x＝性能情報設定回路９の性能情報
y＝性能情報設定回路10の性能情報
S402
イネーブルが全て有効か
No
Yes
S403
無効状態の性能情報設定回路に、有効情報、生成するタスクＩＤ情報及び性能情報を設定する
S404
性能情報設定回路９及び10から性能情報(x,y)を読み込む
注）
S405
MIN(x,y) ＜ 起動したタスクの性能情報
Yes
NO
S406
低い性能情報の性能情報設定回路に、有効情報、生成するタスクＩＤ情報及び性能情報を設定する
S407
タスク管理テーブル15において、起動したタスクに関するペンディングフラグを有効にする
S408
書きかえた性能情報設定回路のタスクＩＤに対応するタスクが起動されているとき、そのペンディングフラグを有効にする
終　了

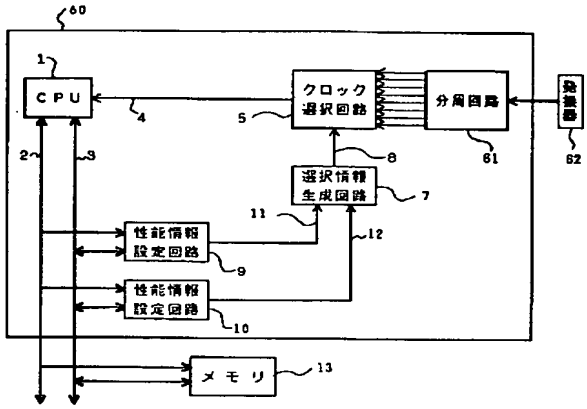
【図６】

図　６



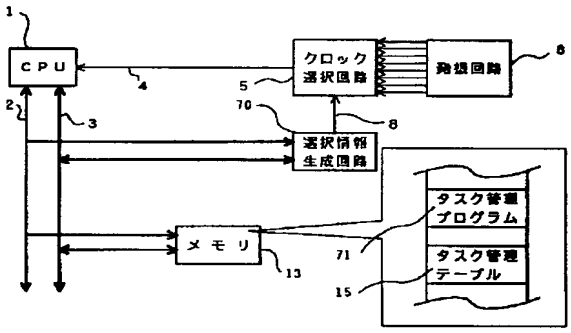
【図７】

図　７



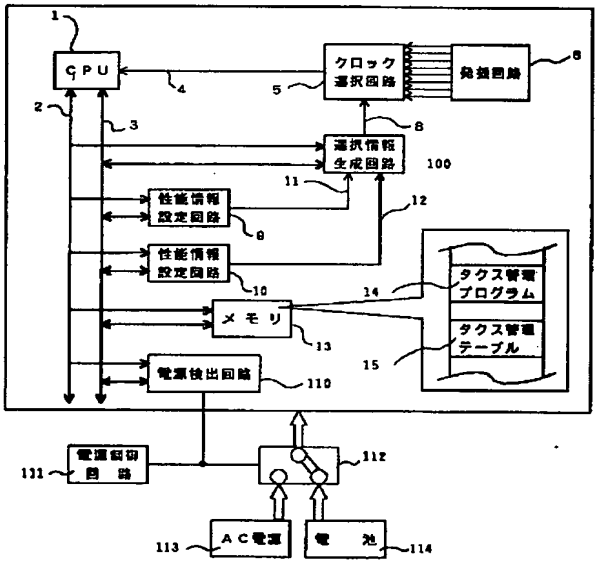
【図８】

図　８



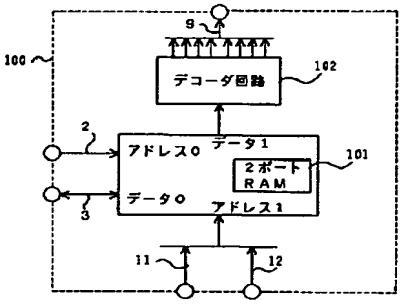
【図１１】

図　11



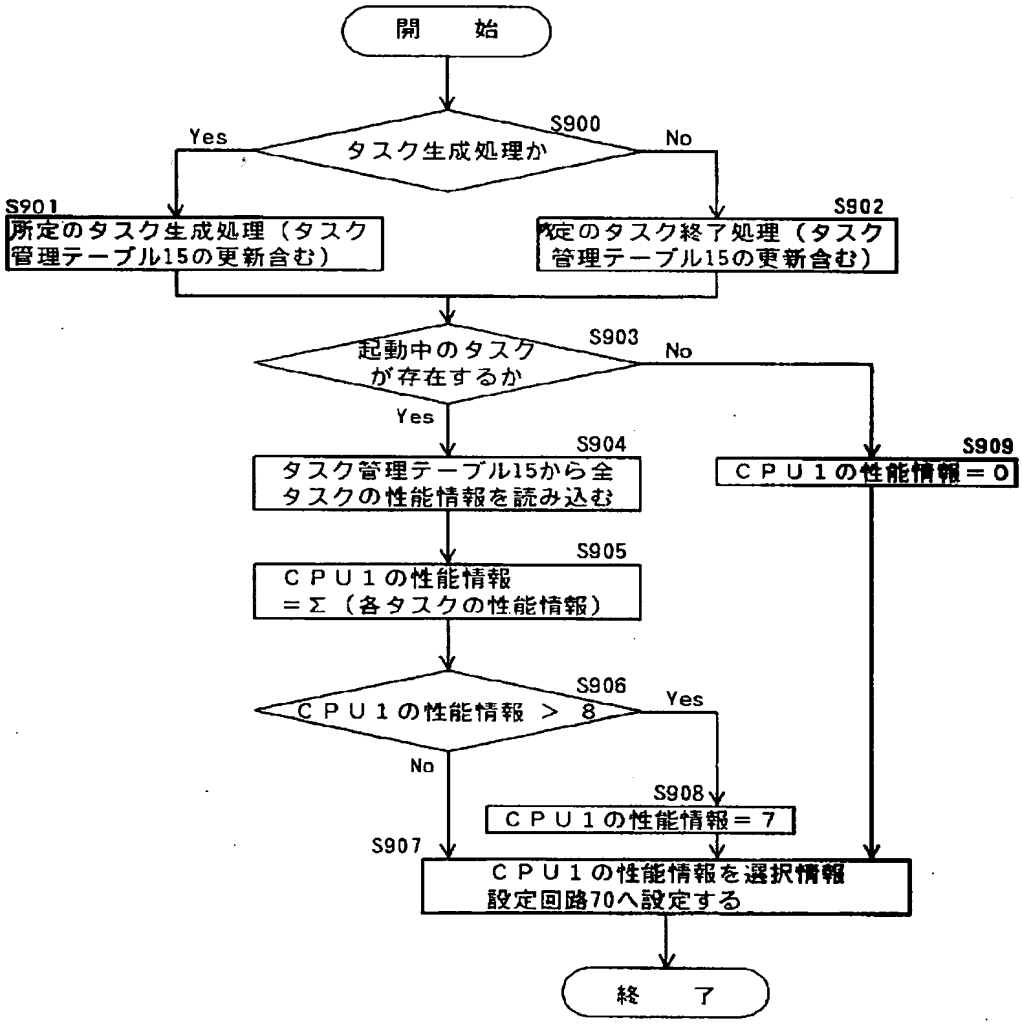
【図１２】

図　12



【図１０】

図　10



フロントページの続き

(72)発明者　大場　信弥
東京都小平市上水本町五丁目20番１号　株
式会社日立製作所半導体事業部内